From Fragmented to Holistic: A Methodological Framework for Variability Management in Enterprise Architecture

Ahmed Dehne[1]

[1] *University of Rostock, Albert-Einstein-Str. 22, 18059 Rostock, Germany, Supervised by Prof. Kurt Sandkuhl*
ahmed.dehne@uni-rostock.de

Abstract

Digital transformation and the proliferation of artificial intelligence (AI) are intensifying variability across enterprises, spanning business processes, data architectures, applications, and IT infrastructures. While Enterprise Architecture (EA) provides a structured lens for analyzing such interdependencies, existing approaches to variability management remain fragmented and limited to isolated layers. This thesis addresses this gap by developing a methodical and technological framework for managing variability holistically within EA. Guided by Design Science Research, the work integrates insights from a structured literature review and an in-depth industrial case study in the energy and water sector. The resulting prototype methodology leverages modular EA building blocks, extended ArchiMate models, and feature-based variability analysis to support consistent modeling across layers. Enhancements such as integrated tool support, visualization of dependencies, and building block representation ensure both clarity and practical usability. Validation is pursued through a dual strategy: academic evaluation in controlled teaching environments to assess comprehensibility and industrial applications across multiple domains to test generalizability and practical value. The research contributes a systematic, variability-aware approach to EA management, bridging academic gaps and meeting urgent industrial demands for agile, modular, and data-aware enterprise solutions.

Keywords

Variability, Enterprise Architecture, Enterprise Architecture Management, ArchiMate Metamodel Extension, Feature Modeling, Enterprise Architecture building block.

1. Introduction

Digital transformation, new business models, and the adoption of cyber-physical systems and artificial intelligence (AI) are driving increasing variability within enterprises. This variability spans multiple layers: business processes often exist in numerous variants, requiring adaptations in data architectures, while smart, connected products demand diverse IT services. Studies on digital transformation (e.g., [1], [2], [3]) and AI's organizational impact (e.g., [4], [5]) show that managing such variability has become both routine and complex. Enterprises typically adopt strategies ranging from strict standardization, which minimizes variability, to highly flexible approaches that allow broad adaptation. In either case, understanding how changes propagate across enterprise layers is essential for effective variability management.

Enterprise Architecture (EA) models support this by visualizing interdependencies across business, data, application, and infrastructure layers. Yet current research and practice provide limited means to manage variability cohesively across these layers. At the same time, the rise of data-driven products and AI applications has increased the importance of data engineering [6]. From an EA perspective, data engineering should align with data architecture, but conflicting priorities often

[1] PoEM2025: Companion Proceedings of the 18th IFIP Working Conference on the Practice of Enterprise Modeling: PoEM Forum, Doctoral Consortium, Business Case and Tool Forum, Workshops, December 3-5, 2025, Geneva, Switzerland
ahmed.dehne@uni-rostock.de

arise: while data engineering focuses on analytics readiness, business process management emphasizes efficiency, resource optimization, and process quality.

Such divergences can create inefficiencies and structural inconsistencies. A potential way forward is tighter integration of business and data architectures, for example by designing modular, data-aware building blocks within processes. This promises adaptability while maintaining control over architectural complexity.

2. Research Objective and Methodological Framework

The objective of our research is to enhance the understanding of variability management in enterprise architecture. The study is conducted within the paradigm of Design Science Research (DSR) [7] , which focuses on solving organizational problems through the design of artefacts—valid and reliable solutions to practical challenges. In this work, the artefact is a methodical and technological framework for managing variability using enterprise architecture (EA) building blocks. DSR projects typically follow four phases: (1) problem investigation, (2) requirements definition, (3) artefact design and evaluation, and (4) artefact demonstration and validation. In this Design Science Research (DSR) study, the artefact is clearly defined as a methodological framework combining conceptual, procedural, and technological dimensions, designed to systematically manage variability in enterprise architectures. For our research, we relied on various theoretical foundations, which can be outlined in the following:

2.1. Enterprise Architecture Management (EAM)

Modern enterprises comprise diverse stakeholders involved in development, operation, and governance, requiring multiple perspectives on structures, processes, and resources. Architectural thinking addresses this by focusing on fundamental enterprise elements, their relationships, and guiding principles [8]. EAM provides a systematic approach to modeling and managing these elements to support planning, transformation, and continuous improvement [9]. The resulting EA acts as a structured map of system states and interdependencies [10]. Among EAM frameworks, TOGAF is widely recognized [11], distinguishing business, information (data and application), and technology architectures. Modeling languages such as ArchiMate further enable consistent representation and communication among stakeholders.

2.2. Variability Management

Variability modeling, originating in software product line engineering [12], addresses the challenge of providing flexible yet maintainable systems. It captures commonalities and differences across system variants, particularly through feature models. A feature is a “distinctive and user-visible aspect, quality, or characteristic of a software system” [13]. Feature models represent commonality (shared properties) and variability (configurable elements) in hierarchical structures called feature trees, visualized in feature diagrams. These diagrams use relations such as mandatory, optional, alternative, and mutually exclusive. Comparative analyses [14], [15] highlight the lack of a universal development method, with approaches often tailored to specific application domains.

2.3. Method Engineering (ME)

ME focuses on designing and adapting methods, tools, and techniques for information systems development [16]. Situational Method Engineering (SME) [17] emphasizes tailoring methods to project context using modular building blocks. Goldkuhl et al [18]. define methods through four elements: (1) components—concepts, procedures, and notation; (2) framework—relations and sequencing; (3) cooperation—roles and responsibilities; and (4) perspective—guiding viewpoint and priorities. This thesis adopts the Goldkuhl framework to structure the proposed method. Throughout this version, the terms 'method' and 'methodology' have been unified under 'methodological framework' for consistency.

3. Why Variability Management in Enterprise Architectures Matters

The motivation for this research arises from both the limited academic coverage of variability management in enterprise architectures and the pressing challenges observed in industrial practice.

From a research perspective, a Structured Literature Review (SLR) was conducted in our prior publication [19] to assess the current state of knowledge in this domain, following Kitchenham's methodology [20]. The review focused on the question:

RQ: What is the state of research on managing variability in enterprise architectures?

The findings reveal that variability management in Enterprise Architecture Management (EAM) is still underexplored. Only a small number of studies address this topic, and most focus on isolated aspects or specific architectural layers. Existing approaches extend EA metamodels at the business process level, apply software product line techniques to application or technology architectures, or introduce specialized modeling approaches for data and service layers. While these contributions provide valuable insights, they remain fragmented. No comprehensive method was identified that addresses variability management consistently across all EAM layers. This academic gap underscores the need for research into integrated frameworks capable of systematically managing variability throughout the enterprise architecture. Throughout this version, the terms 'method' and 'methodology' have been unified under 'methodological framework' for consistency.

The industrial perspective reinforces this need. An in-depth case study with a leading solution provider in the energy and water sector highlighted the challenges of managing variability in practice. The company develops software that spans business, application, data, and technology layers, documented in ArchiMate models. These models are used both to define product roadmaps and to create client-specific instantiations. However, client solutions often go beyond simple configurations, requiring tailored combinations of functionalities. The company's ambition to assemble customer solutions through reusable building blocks exposes the difficulty of decomposing enterprise architectures into modular yet coherent elements. The challenge is twofold: ensuring that each building block includes only the minimum required data and functionality while at the same time avoiding excessive fragmentation of the overall architecture. Dependencies across activities, data, and services must be explicitly captured, for instance through feature-based modeling techniques. Moreover, the growing integration of artificial intelligence—used for anomaly detection, churn management, and predictive analytics—further complicates the design of data architectures, which must remain both flexible and consistent to satisfy integrity requirements.

Together, these findings highlight a critical research opportunity. On the one hand, academia has yet to provide holistic methods for variability management in enterprise architectures. On the other, industry urgently requires systematic approaches to modularize architectures and enable agile, variability-aware solution development. This dual perspective—academic gap and industrial demand—forms the foundation and justification for the research presented in this thesis.

4. Method Requirements and Prototype Method for Variability Management in Enterprise Architecture

This chapter outlines the principles guiding the definition of method requirements and introduces a systematic method for managing variability in Enterprise Architecture (EA). The approach draws insights from both the literature review and the industrial case study. The method supports modular, variability-aware modeling across the business, application, and data layers of EA, using standardized modeling languages and tools. The data layer denotes the architectural level that captures data entities, their relationships, and information flows linking business processes and applications. In ArchiMate [21], it aligns with the Information system layer and serves as a bridge between business and application architectures, ensuring that data variability is modeled consistently with functional and operational aspects.

4.1. Method Principles

The development of a Method for managing variability in EA follows principles from Method Engineering. A method must:

- Be requirements-driven, addressing gaps from both academic literature and industrial practice.
- Support modularity, ensuring that complex architecture can be decomposed into coherent building blocks.
- Ensure standardization, using widely accepted modeling languages and tools to foster communication among stakeholders.
- Enable visualization, providing clear representations of dependencies, options, and variation points across layers.
- Remain practical, by aligning with existing commercial tool support instead of requiring entirely new implementations.

These principles guided both the identification of requirements and the design of the prototype method.

4.2. Method Requirements

Based on the structured literature review and the industrial case study, we derived six key requirements for method support (summarized in Table 1).

Table 1
Method Support Requirements

No.	Requirement	Category	Description
1	Address variability across EA layers	Literature	Variability management must extend beyond isolated layers to business, application, and data.
2	Provide a framework for combining approaches	Literature	Different methods must be integrated into a cohesive framework with practical guidelines.
3	Support integrated modeling	Case study / Literature	Unified modeling across all EA layers is necessary; existing approaches remain fragmented.
4	Define building blocks in standardized languages	Case study	Standardized modeling (e.g., ArchiMate) ensures communication and reuse across stakeholders.
5	Leverage existing commercial tool support	Case study	Tools must be available and cost-effective, avoiding proprietary or custom-built solutions.
6	Visualize dependencies and configuration options	Case study	Variability and dependencies must be explicitly represented to reduce errors and aid decisions.

4.3. Prototype Method

The proposed prototype Method builds on the concept of building blocks as reusable parts of EA. Each building block represents a modular unit, typically including one or more business processes, supporting application and data elements, and explicit interfaces for integration.

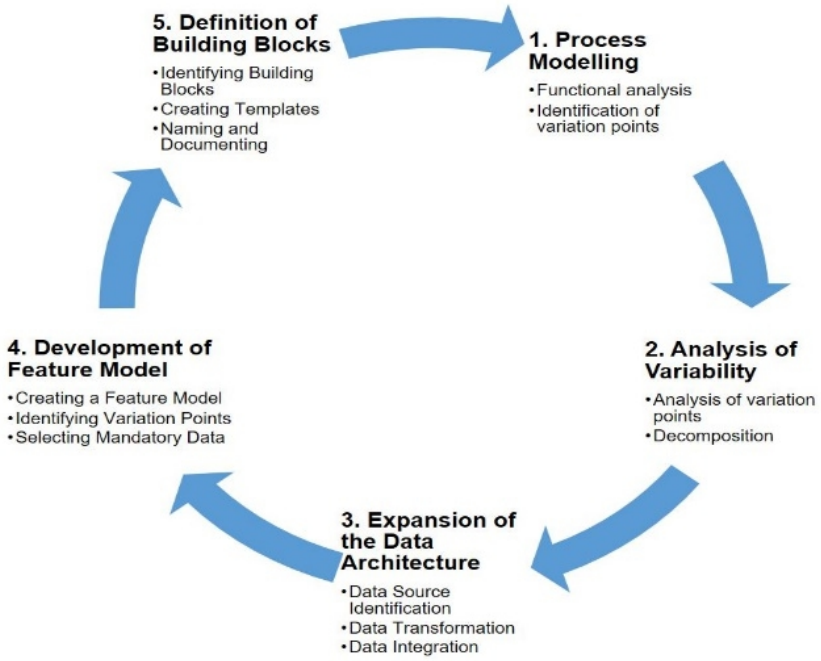

Figure 1: Method approach and framework of the method prototype

The prototype method proceeds through five systematic steps:

- Process Modeling
    a. Analyze processes to identify activities, actors, data flows, and resources.
    b. Detect variation points where process variants occur.
- Variability Analysis
    a. Classify variation points (mandatory, optional, alternative).
    b. Decompose processes into modules aligned with variability patterns.
- Expansion of Data Architecture
    a. Identify relevant data sources.
    b. Transform and integrate data into process models.
- Feature Model Development
    a. Create feature diagrams linking processes and data.
    b. Represent mandatory, optional, and alternative variation points.
    c. Tailor process models to reflect selected data and configuration options.
- Definition of Building Blocks
    a. Identify modular building blocks combining processes, applications, and data.
    b. Represent blocks as ArchiMate models and link them to feature models.
    c. Document blocks for reuse across architectural contexts.

This structured approach establishes a repeatable procedure that satisfies the requirements identified in Section 4.2.

4.4. Method Enhancements

During implementation and validation in the industrial case study, several enhancements were introduced:

- Integration of ArchiMate and Feature Modeling: To overcome fragmentation, ArchiMate was extended with feature-modeling constructs (mandatory, optional, alternative features) to represent variability directly within EA models.
- Unified Tool Support: Instead of relying on separate environments (e.g., Archi and PowerPoint), feature modeling was embedded in Archi, eliminating synchronization overhead and reducing the risk of inconsistencies.
- Building Block Representation: A specialized extension of ArchiMate (via Group and Plateau elements) was introduced to represent building blocks as modular units, enabling traceability across layers.
- Clear Visualization of Dependencies: Feature diagrams were incorporated into EA views to explicitly show dependencies between processes, applications, and data objects.

These enhancements improved the efficiency, maintainability, and clarity of EA variability models, allowing organizations to construct solutions by recombining building blocks while ensuring architectural integrity.

5. Validation Strategies and Future Application

A core component of method development in Design Science Research (DSR) [7] is validation, which ensures that the proposed method is both theoretically sound and practically effective. To this end, two complementary validation strategies are proposed: one situated in an academic environment and the other in industrial contexts, enabling iterative refinement through controlled experimentation and real-world application. This section refines these strategies by outlining measurable indicators such as clarity, usability, time-to-complete, correctness, and perceived acceptance. Academic validation will rely on controlled experiments comparing standard ArchiMate with the extended feature-aware version. For industrial evaluation, future studies will consider Technology Acceptance Model (TAM) [22] and Unified Theory of Acceptance and Use of Technology (UTAUT) [23] as potential frameworks for assessing practitioner acceptance and usefulness, acknowledging that these remain strategic suggestions requiring further definition and practical testing

5.1. Academic Validation

The first validation strategy focuses on applying the method in a controlled academic setting. The method will be documented and taught as part of a practical course module at the university. A simulated case will be prepared and provided to students, who will be asked to apply the method to analyze, model, and manage variability in enterprise architectures.

During this exercise, student interactions with the method will be carefully observed, and specific attention will be paid to:

- Ease of understanding: clarity of method components and modeling steps.
- Practical usability: ability of students to apply the method with limited prior exposure.
- Obstacles and challenges: points where students struggle, misunderstand, or require additional explanation.
- Outcome quality: completeness and correctness of the student-generated EA models and building blocks.

The observations will be systematically documented and analyzed to identify recurring challenges. These findings will then feed back into the development process of the method, allowing for

incremental refinement of requirements, concepts, and supporting materials (e.g., guidelines, tool instructions). In this way, the academic validation serves as a formative evaluation, improving accessibility and pedagogical clarity.

5.2. Industrial Validation

The second validation strategy involves applying the method in industrial case studies. While the initial application was conducted in the energy and water sector, further studies will extend to other domains, such as finance, manufacturing, and telecommunications. These domains are chosen because they exhibit complex enterprise architectures, significant variability across business processes, and increasing reliance on digital transformation and modular system design.

By applying the method in diverse industrial contexts, it will be possible to:

- Test the generalizability of the method across different industries.
- Assess the practical value of metamodel extensions and building block representations under real-world conditions.
- Identify domain-specific requirements that may not have been evident in the energy and water case study.

Refine both the method requirements and the method prototype through iterative feedback loops. The outcome of these industrial applications will be a more robust, flexible, and industry-aligned methodology for variability management in enterprise architecture.

5.3. Iterative Refinement

Both academic and industrial validations are intended to function as iterative refinement mechanisms. Academic validation ensures that the method is understandable, teachable, and practically usable by individuals with varying levels of expertise. Industrial validation, in contrast, ensures that the method provides real value in complex, dynamic, and domain-specific settings.

Through this dual approach, the method can evolve in a balanced way: academically rigorous, practically relevant, and broadly applicable across different enterprise contexts. Ultimately, these validation strategies will help ensure that the method not only addresses theoretical gaps but also delivers tangible benefits to both researchers and practitioners.

Declaration on Generative AI

During the preparation of this work, the author(s) used ChatGPT-5 in order to: Grammar and spelling check. After using these tool(s)/service(s), the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] Kaidalova J, Sandkuhl K, Seigerroth U (2018) How digital transformation affects enterprise architecture management-a case study. International Journal of Information Systems and Project Management 6:5–18

[2] Zimmermann A, Schmidt R, Sandkuhl K et al. (2018) Evolution of enterprise architecture for digital transformation. In: 2018 IEEE 22nd International Enterprise Distributed Object Computing Workshop (EDOCW), pp 87–96

[3] Sandkuhl K, Seigerroth U (2021) Digital Transformation of Enterprises: Case Studies and Transformation Paths. In: PACIS, p 35

[4] Rittelmeyer JD, Sandkuhl K (2021) Effects of artificial intelligence on enterprise architectures-a structured literature review. In: 2021 IEEE 25th International Enterprise Distributed Object Computing Workshop (EDOCW), pp 130–137
[5] Park S, yoon Lee J, Lee J (2024) AI system architecture design methodology based on IMO (Input-AI Model-Output) structure for successful AI adoption in organizations. Data & Knowledge Engineering 150:102264
[6] Fruhwirth M, Ropposch C, Pammer-Schindler V (2020) Supporting Data-Driven Business Model Innovations: A structured literature review on tools and methods. Journal of Business Models 8:7–25
[7] Hevner A, Chatterjee S (2010) Design research in information systems: theory and practice, vol 22. Springer Science & Business Media
[8] Winter R (2014) Architectural thinking. Wirtschaftsinformatik 56:395–398
[9] Simon D, Fischbach K, Schoder D (2014) Enterprise architecture management and its role in corporate strategic management. Information Systems and e-Business Management 12:5–42
[10] Ahlemann F, Stettiner E, Messerschmidt M et al. (2012) Strategic enterprise architecture management: challenges, best practices, and future developments. Springer Science & Business Media
[11] The Open Group (2022) The TOGAF® Standard, 10th Edition: Architecture Development Method. Van Haren Publishing
[12] Barth B, Butler G, Czarnecki K et al. (2002) Generative programming. In: Object-Oriented Technology: ECOOP 2001 Workshop Reader: ECOOP 2001 Workshops, Panel, and Posters Budapest, Hungary, June 18-22, 2001 Proceedings 15, pp 135–149
[13] Kang KC, Cohen SG, Hess JA et al. (1990) Feature-oriented domain analysis (FODA) feasibility study
[14] Thörn C, Sandkuhl K (2009) Feature modeling: managing variability in complex systems. Complex Systems in Knowledge-based Environments: Theory, Models and Applications:129–162
[15] Li L, Zheng Y, Yang M et al. (2020) A survey of feature modeling methods: Historical evolution and new development. Robotics and Computer-Integrated Manufacturing 61:101851
[16] Brinkkemper S (1996) Method engineering: engineering of information systems development methods and tools. Information and software technology 38:275–280
[17] Henderson-Sellers B, Ralyté J, Ågerfalk PJ et al. (2014) Situational method engineering
[18] Goldkuhl G, Lind M, Seigerroth U (1998) Method integration: the need for a learning perspective. IEE Proceedings-Software 145:113–118
[19] Dehne A, Sandkuhl K (2023) Variability modelling in enterprise architecture management-survey on existing approaches. In: 22nd International Conference on Perspectives in Business Informatics Research Workshops and Doctoral Consortium, BIR-WS 2023 Ascoli Piceno 13 September 2023 through 15 September 2023, vol 3514, pp 94–107
[20] Kitchenham B, Brereton OP, Budgen D et al. (2009) Systematic literature reviews in software engineering-a systematic literature review. Information and software technology 51:7–15
[21] Group TO ArchiMate 3.2 Specification
[22] Davis FD, Granić A (2024) The Technology Acceptance Model. Springer International Publishing, Cham